

This Page Is Inserted by IFW Operations  
and is not a part of the Official Record

## **BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning documents *will not* correct images,  
please do not report the images to the  
Image Problem Mailbox.**

Requested Patent: EP1004964A1

Title:

DEVICE AND METHOD FOR OPTIMIZATION OF MONITORING THRESHOLDS ;

Abstracted Patent: EP1004964 ;

Publication Date: 2000-05-31 ;

Inventor(s): BONNIN ALAIN (FR) ;

Applicant(s): BULL SA (FR) ;

Application Number: EP19990402940 19991126 ;

Priority Number(s): FR19980014936 19981127 ;

IPC Classification: G06F11/34 ;

Equivalents: FR2786581

**ABSTRACT:**

The performance agent (2i) executes at least a diagnosis check when a step has been exceeded by calling up an internal diagnostic function (5i) of the SNMP performance agent (2i) managed by the SNMP(1) of the local computer (OL), and to transmit a diagnostic notification including diagnostic data. The device is designed to permit monitoring of remote computers (OD1, ... Odn) from an SNMP manager (1) installed in a local computer (OL) through a network (10). The device includes a performance agent (2) installed in each remote computer (Odi), designed to detect the passing of a consumption step or several resources of the remote computer (Odi) from a function of the step (8i), managed by the SNMP (1) of the local computer (OL). This function allows parameter setting of the performance agent by indicating a determined step for each monitored activity and transmission of any passing of the step level. An Independent Claim is included for a method for optimizing steps for consumption parameters.



Europäisches Patentamt  
European Patent Office  
Office européen des brevets



(11) **EP 1 004 964 A1**

(12) **DEMANDE DE BREVET EUROPEEN**

(43) Date de publication:  
31.05.2000 Bulletin 2000/22

(51) Int Cl.<sup>7</sup>: **G06F 11/34**

(21) Numéro de dépôt: **99402940.3**

(22) Date de dépôt: **26.11.1999**

(84) Etats contractants désignés:  
**AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU  
MC NL PT SE**  
Etats d'extension désignés:  
**AL LT LV MK RO SI**

(71) Demandeur: **BULL S.A.**  
**78434 Louveciennes Cedex (FR)**

(72) Inventeur: **Bonnin, Alain**  
**95150 Taverny (FR)**

(30) Priorité: **27.11.1998 FR 9814936**

(54) **Dispositif et procédé d'optimisation de surveillance de seuils**

(57) La présente invention concerne un dispositif et un procédé d'optimisation de surveillance de seuils. Le dispositif selon l'invention permettant de surveiller l'activité d'au moins un ordinateur distant (OD1, ODn) comprend un agent de performance (2i), à partir d'un gestionnaire SNMP (1) installé sur un ordinateur local (OL) à travers un réseau (10), ledit agent de performance (2i) pouvant détecter le dépassement d'un seuil de consommation d'une ou plusieurs ressources de l'ordinateur distant (OD) à partir d'une fonction de seuil (8i) qui permet de paramétrer l'agent de performance en lui indiquant les seuils déterminés pour chaque activité surveillée, et émettre une notification vers le gestionnaire SNMP (1) de l'ordinateur local (OL) suite au dépassement d'un seuil. L'agent de performance (2i) exécute au moins un diagnostic lorsqu'un seuil a été dépassé en appelant une fonction de diagnostic (5i) interne à l'agent SNMP de performance (2i) et émet une notification comprenant l'information de diagnostic donnant des informations précises sur les origines de la consommation des ressources surveillées au moindre coût.

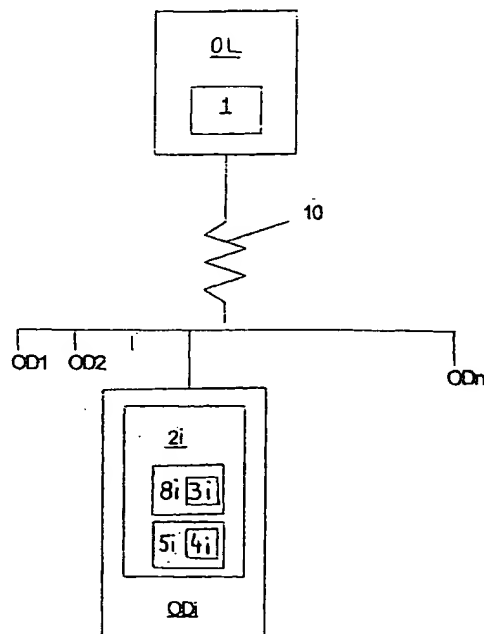


Figure 1

EP 1 004 964 A1

## Description

[0001] La présente invention concerne un dispositif et un procédé d'optimisation de surveillance de seuils.

[0002] La présente invention concerne plus particulièrement un dispositif et un procédé permettant de surveiller l'activité d'au moins un ordinateur distant comprenant un agent de performance, à partir d'un gestionnaire SNMP installé sur un ordinateur local, à travers un réseau.

[0003] Il existe aujourd'hui un réseau mondial d'ordinateurs qui se nomme Internet. Le réseau Internet repose sur une famille de protocoles nommée "TCP/IP". Pour administrer un ordinateur sur le réseau Internet à partir d'un autre ordinateur, on utilise le protocole SNMP (Simple Network Management Protocol) de la famille des protocoles TCP/IP. C'est, en informatique, un protocole d'administration standardisé par des documents appelés "RFC" (Request For Comments). Lorsque le réseau fonctionne en mode non connecté, le protocole est de type UDP/IP. Le protocole SNMP permet à un ordinateur ci-après dénommé "l'ordinateur local" (OL), comportant un logiciel de gestionnaire SNMP, ci après appelé "le gestionnaire SNMP", de communiquer, par un protocole réseau de type SNMP/UDP/IP, avec un autre ordinateur, ci après appelé "l'ordinateur distant" (OD), comportant un logiciel d'agent SNMP, ci dessous appelé l'agent SNMP. Pour s'adresser sur un réseau à un agent SNMP, à partir du gestionnaire SNMP, on utilise un port UDP et une adresse IP. Un agent est un programme actif ou processus qui fournit un service. Ce processus peut se décomposer en plusieurs unités d'exécution appelée également "thread".

[0004] Il existe de nombreux agents SNMP, sur les ordinateurs, sur les routeurs, etc. Le dialogue SNMP de ces agents est constitué de commandes et d'un message rudimentaires, respectivement "GET", "GET NEXT", "SET" et "TRAP". C'est pour cette raison que le protocole est qualifié de simple. Les commandes "GET", "GET NEXT" et "SET" sont des requêtes émises par le gestionnaire SNMP. Les messages d'alarme (TRAP) sont des événements émis par l'agent SNMP de l'ordinateur distant (OD) et non sollicités par le gestionnaire SNMP de l'ordinateur local (OL).

[0005] Les objets SNMP sont ordonnés par leurs identificateurs (OID). Les ordinateurs étant de plus en plus puissants, le nombre des objets, imprimantes ou fichiers par exemple, gérés par les agents est de plus en plus grand. Le protocole UDP n'est pas sécurisé. Une trame UDP peut se perdre sur le réseau sans que l'émetteur en soit averti. Le gestionnaire SNMP doit en cas de non-réponse, refaire sa requête après un délai d'attente.

[0006] Un agent SNMP gère une MIB (Management Information Base - base d'information managée) SNMP qui se décompose en objets SNMP organisés sous forme arborescente. Par exemple, il peut gérer un objet SNMP global et unique comme le pourcentage de consommation globale du ou des processeurs de l'ordinateur ou bien des objets SNMP multiples comme le pourcentage de consommation de CPU d'une application, ce qui entraîne qu'il y a autant d'instances gérées par l'agent SNMP de ce type d'objet SNMP qu'il y a d'applications sur l'ordinateur. Dans ce dernier cas, l'objet est dit multi-instancié et est représenté par une « table SNMP ». L'agent SNMP qui gère la consommation des ressources sur un ordinateur est appelé ci-après "l'agent SNMP de performance".

[0007] Pour surveiller l'activité d'un ordinateur à partir d'un gestionnaire SNMP d'un ordinateur local, à travers un réseau, une solution consiste à fabriquer un agent SNMP de performance dans l'ordinateur distant. Le gestionnaire SNMP émet ensuite des requêtes SNMP "CET" et "GET NEXT" vers cet agent distant SNMP de performance pour obtenir périodiquement par exemple la liste des programmes actifs et leur pourcentage de consommation de ressources. Cette solution est très coûteuse en argent car les requêtes réseau sont facturées à la quantité ou au temps d'utilisation du réseau. Cette solution est également très coûteuse en charge car elle entraîne de nombreuses requêtes réseau, surtout si ces dernières sont à destination d'un ordinateur lui-même très chargé.

[0008] Le but de l'invention est d'optimiser, à partir d'un gestionnaire SNMP, la surveillance à travers un réseau, de l'activité d'un ordinateur distant, par un agent SNMP et de proposer un dispositif qui permette à une personne chargée de la surveillance d'un réseau d'ordinateurs d'être prévenue lorsque l'utilisation des ressources d'un ordinateur dépasse un certain seuil et de connaître quels sont les programmes actifs sur cet ordinateur qui ont entraîné ce dépassement de seuil tout en augmentant le moins possible la consommation des ressources de cet ordinateur.

[0009] Ce but est atteint par le fait que le dispositif permettant de surveiller l'activité d'au moins un ordinateur distant, à partir d'un gestionnaire SNMP installé sur un ordinateur local à travers un réseau, se caractérise en ce qu'il comprend un agent de performance installé dans chaque ordinateur distant pouvant détecter le dépassement d'un seuil de consommation d'une ou plusieurs ressources de l'ordinateur distant à partir d'une fonction de seuil, gérable par le gestionnaire SNMP de l'ordinateur local, fonction qui permet de paramétrer l'agent de performance en lui indiquant des seuils déterminés pour chaque activité surveillée, et émettre une notification de dépassement de seuil, et en ce que l'agent de performance exécute au moins un diagnostic de seuil lorsqu'un seuil a été dépassé en appelant une fonction de diagnostic interne à l'agent SNMP de performance, fonction gérable par le gestionnaire SNMP de l'ordinateur local, et émet une notification de diagnostic comprenant une information de diagnostic.

[0010] Selon une autre particularité, les paramètres de la fonction interne de seuil et de diagnostic sont des indexes indirects d'accès à une table respectivement de seuil et de diagnostic de l'agent de performance.

[0011] Selon une autre particularité, la table de seuil est constituée des objets suivants : un identificateur, une formule, une période, un type de comparaison, une valeur à comparer (seuil), un nombre d'occurrence avant alerte, une sévérité,

un nombre d'alarmes possibles, une fonction interne de diagnostic à appeler en cas de dépassement de seuil, une date, une valeur de fin d'alerte, une période d'activation.

[0012] Selon une autre particularité, la table de diagnostic est constituée des objets suivants : un identificateur, une formule, une période, un type de comparaison, une valeur à comparer, une sévérité, une date.

5 [0013] Selon une autre particularité, la table de seuil et la table de diagnostic comportent un objet définissant la destination de la notification de dépassement de seuil et un objet définissant la destination de la notification de diagnostic.

[0014] Selon une autre particularité, la notification est constituée d'un message d'alarme (TRAP) ou d'une écriture dans un historique de son propre programme ou dans un historique global du système.

10 [0015] Selon une autre particularité, chaque activité surveillée est effectuée par une unité d'exécution qui déclenche une autre unité d'exécution pour chaque diagnostic.

[0016] Selon une autre particularité, la fonction de diagnostic (5i) comporte quatre paramètres, permettant de déclencher respectivement le calcul de la consommation globale en processeur, de la mémoire physique, de la mémoire du cache, du réseau, un cinquième paramètre permettant de déclencher la recherche de tous les programmes qui  
15 utilisent plus d'un pourcentage paramétrable du ou des processeurs, un sixième paramètre permettant de déclencher le calcul ces consommations sur un délai paramétrable en millisecondes, et un septième paramètre permettant de déclencher la recherche de tous les programmes qui utilisent plus d'un nombre paramétrable de kilo octets de mémoire physique.

[0017] Un autre but de l'invention est de proposer un procédé d'optimisation de surveillance de seuils de consommation paramétrables sur un ordinateur distant.

[0018] Ce but est atteint par le fait que le procédé comporte les étapes suivantes :

- une étape de surveillance d'un seuil par une fonction interne de seuil;
- une étape d'émission, par l'ordinateur distant sur lequel l'agent de performance effectue la surveillance, d'une  
25 notification de dépassement de seuil;
- une étape d'exécution de diagnostics par une fonction de diagnostic interne ;
- une étape d'émission d'une notification de diagnostic contenant le diagnostic associé au dépassement de seuil.

[0019] Selon une autre particularité, l'étape de surveillance de seuil comporte trois tâches, une première d'analyse d'une formule de la fonction de seuil, une deuxième d'obtention à un temps donné des valeurs des compteurs d'une première structure interne et une troisième d'exécution d'une boucle infinie de surveillance.

[0020] Selon une autre particularité, la boucle infinie de surveillance comprend une première partie consistant à attendre sur une période et à obtenir au bout de la période les valeurs des compteurs d'une seconde structure, une deuxième partie consistant à tester à l'aide des deux structures internes si un seuil de la fonction de seuil a été atteint  
35 et si le nombre d'occurrences est atteint, et une troisième partie consistant à remplacer les valeurs des compteurs de la première structure interne par ceux de la seconde.

[0021] Selon une autre particularité, l'exécution d'un diagnostic comporte l'analyse de la formule en préparant les valeurs des compteurs de diagnostic de deux structures internes, la recherche du nombre d'occurrence des compteurs de diagnostic et l'exécution une boucle de surveillance pour chaque occurrence.

40 [0022] Selon une autre particularité, la boucle comprend une première partie consistant à trouver, à un temps donné, la valeur des compteurs de la première structure interne, afin d'initialiser le diagnostic, une deuxième partie consistant à attendre sur sa périodicité, puis à obtenir sur ce nouveau temps donné les valeurs des compteurs de surveillance de la seconde structure interne et une troisième partie consistant à tester à l'aide des deux structures si le seuil a été atteint.

45 [0023] D'autres particularités et avantages de la présente invention apparaîtront plus clairement à la lecture de la description ci-après faite en référence aux figures annexées suivantes :

- la figure 1 représente une vue schématique d'un système de surveillance à partir d'un gestionnaire SNMP de l'activité d'un ordinateur par un agent SNMP de performance ;
- 50 - les figures 2 et 3 représentent respectivement une table de seuil et une table de diagnostic ;
- les figures 4 et 5 représentent une structure interne d'une unité d'exécution respectivement d'une surveillance de seuil et d'un diagnostic de seuil.

[0024] En référence à la figure 1, un ordinateur local (OL) est équipé d'un logiciel gestionnaire SNMP (1) et est  
55 connecté à un réseau (10), par exemple Internet, et à travers ce réseau (10), à des ordinateurs distants (OD1, OD2, ..., OD<sub>i</sub>, ..., OD<sub>n</sub>). Chaque ordinateur distant (OD<sub>i</sub>) gère des applications et comporte un agent SNMP de performance (2i). Cet agent de performance est un programme actif ou processus qui surveille la consommation globale des ressources de l'ordinateur distant (OD<sub>i</sub>). Chaque machine administrée comporte un agent de performance (2i) muni d'une

fonction (8i) de seuil permettant de surveiller des paramètres tels que l'activité CPU, l'activité machine, les programmes actifs, les disques, l'activité mémoire, et l'activité des protocoles sur le réseau. La fonction de seuil (8i) peut être paramétrée en lui indiquant de surveiller des seuils et de manière avantageuse des seuils globaux portant sur plusieurs ressources. Des seuils globaux sont déterminés pour chacune des activités surveillées. L'agent de performance (2i) comporte une première table de paramétrage de seuil appelée "table de seuil" ou "table d'alarme" (3i) dans laquelle sont stockés des seuils associés à des combinaisons d'attributs et de formules. Les combinaisons de seuils, d'attributs et de formules constituent des requêtes de surveillance. Chaque argument de la fonction de seuil correspond à une requête de surveillance. Les seuils, formules et attributs consistent en des objets SNMP. L'agent de performance crée une unité d'exécution par requête de surveillance, et plus précisément par point d'entrée de la table de seuil. Si la consommation d'une activité surveillée dépasse un seuil de la table de seuil soit sur dépassement d'une consommation globale de toutes les ressources de l'ordinateur, soit sur dépassement d'une consommation d'une ou plusieurs ressources, soit par n'importe quelle requête de surveillance de dépassement de seuil, ou encore si une période de vérification est atteinte, l'agent de performance émet une notification qui est constituée soit d'un message d'alarme (TRAP) vers le gestionnaire SNMP (1), soit d'une écriture dans un historique de son propre programme ou dans un historique global du système.

**[0025]** Lorsqu'une notification est émise, l'agent de performance appelle une fonction interne de diagnostic (5i) associée à ladite notification. La fonction interne de diagnostic est destinée à diagnostiquer le dépassement de seuil, à savoir rechercher l'origine du dépassement, effectuer un traitement en conséquence et/ou surveiller la ressource l'ayant occasionné. Cette fonction est configurable avec des paramètres. L'agent de performance (2i) comporte une deuxième table de paramétrage de diagnostic (4i) appelée "table de diagnostic". Chaque argument de la fonction de diagnostic correspond à un point d'entrée de la table de diagnostic, et donc à une requête de diagnostic. L'agent de performance crée une unité d'exécution par point d'entrée de la table de diagnostic. Une seconde forme de réalisation consiste à utiliser des fonctions internes de seuil (8i) et de diagnostic (5i) avec des arguments prédéfinis sans passer par des tables. Dans la première forme de réalisation, les types de diagnostic sont susceptibles d'être enrichis en créant de nouvelles requêtes de diagnostic.

**[0026]** C'est l'agent de performance lui-même qui envoie la consommation en ressources de chaque programme actif par un message d'alarme (TRAP) au gestionnaire SNMP. Cette solution est paramétrable. Cette solution n'entraîne aucun coût, car elle ne nécessite pas de requêtes réseau, ni d'appel à un programme externe. Le seul coût est celui généré par l'appel à une fonction interne d'un programme qui est actif.

**[0027]** Une description détaillée d'un agent SNMP de performance conforme à l'invention, va à présent être effectuée. Cet agent SNMP de performance permet d'émettre une notification (message d'alarme ou écriture dans un historique) quand un seuil, associé à une combinaison d'attributs de performance et de formules dite requête de surveillance, est atteint, ou quand une période de vérification est atteinte même s'il n'y a pas de dépassement de seuil.

**[0028]** Les formules et attributs sont stockés dans des objets SNMP, et constituent une table "ntP100Alarm" appelée "table d'alarme" ou "table de seuil" (3i). Une formule et son seuil associé peuvent être modifiés. Pour configurer les seuils, l'agent de performance lance une unité d'exécution ou "thread" du programme actif, en utilisant les objets de la table de seuil (3i) définie ci-après. Tous les attributs des objets de la table de seuil sont de type "chaîne de caractères SNMP" et possèdent une valeur par défaut. Tous les attributs sont optionnels. Il n'existe pas d'attributs obligatoires. La table de seuil "ntP100Alarm" (3i) de l'agent de performance ntPerformance comporte les objets suivant :

**[0029]** Un objet "ntP100AlarmFormula" définit une formule, dont la valeur par défaut est la chaîne :  

$$*100 - (\text{DELTA}(\text{ntP01TotalProcessorTime}) / (\text{DELTA}(\text{ntP01CaptureTime}) * 100000))$$

**[0030]** Cette chaîne représente, par exemple, l'activité CPU globale en pourcentage. L'attribut "ntP01TotalProcessorTime" représente un premier compteur 64 bits incrémental, avec une précision de 100 nanosecondes. L'attribut "ntP01CaptureTime" représente un compteur de temps absolu avec une précision de 1 milliseconde. '100000' est la différence de précision entre ces deux compteurs. La formule peut par exemple contenir des entiers, "+" pour additionner, "-" pour soustraire, "\*" pour multiplier, "/" pour diviser, "(" et ")" pour donner la priorité, des compteurs "ntP..." disponibles dans la MIB "ntPerformance" de l'ordinateur distant. L'opération "DELTA()" calcule la différence entre deux valeurs successives d'un même compteur "ntP". L'attribut ";" après un objet "ntP" et suivi d'un entier indique l'occurrence du compteur "ntP". A titre d'exemple (figure 2), "ntP06FreeSpace:1" représente le premier disque logique, qui est généralement "C:".

**[0031]** Un objet "ntP100AlarmComparisonValue" représente la valeur du seuil. Sa valeur par défaut est 90. La formule est régulièrement calculée par l'agent de performance. Lorsque la formule, représentant l'activité CPU, par exemple, devient supérieure au seuil, par exemple 90%, l'agent peut émettre une notification constituée soit par un message d'alarme "TRAP" soit par une écriture dans un historique.

**[0032]** Un objet "ntP100AlarmID" est un index pour la table de seuil permettant d'identifier chaque requête de surveillance constituant ladite table. Des exemples de surveillance de seuils d'une table de seuil sont définis en annexe 1. Cet objet ntP100AlarmID peut être utilisé pour entrer ou détruire un seuil.

**[0033]** Un objet "ntP100AlarmPeriod" définit la période de surveillance en seconde. Sa valeur par défaut est 3600

(une heure). Par exemple l'activité CPU sera calculée périodiquement, selon cet intervalle spécifié. Si cette activité, en une heure, est supérieure à 90%, une notification est émise.

[0034] Un objet "ntP100AlarmComparisonType" spécifie le type de comparaison, si la valeur effective doit être supérieure (>) ou inférieure (<) à la valeur du seuil pour émettre une notification. Par défaut, c'est la valeur supérieure (>) qui est par exemple choisie.

[0035] Un objet "ntP100AlarmRepeat" est un entier qui représente le nombre de fois successives qu'un seuil de surveillance doit être atteint avant qu'une notification soit émise. La valeur par défaut est par exemple "0". Dans ce cas, la notification est alors immédiatement émise. La valeur "n" signifie que la notification est émise seulement après "n" plus une occurrences consécutives de dépassement de seuil. "n" est un entier positif.

[0036] Un objet "ntP100AlarmSeverity" est une chaîne qui contient le niveau de sévérité des notifications émises. La valeur peut être "tout va bien", "avertissement", "mineur", "majeur" or "critique". Ces valeurs correspondent à des niveaux de sévérité standardisés. Par défaut le niveau est fixé à "avertissement".

[0037] Un objet "ntP100AlarmOnce" est un indicateur. Si cet indicateur est positionné à "vrai", une notification dite notification d'alerte est émise seulement la première fois que le seuil est atteint et la surveillance de seuil n'est plus effectuée. En conséquence, aucune notification de fin d'alerte "tout va bien", définie ci-après, ne peut avoir lieu. Cet indicateur est utile pour conserver le niveau de sévérité du message d'alarme. La valeur par défaut est "faux" et implique que la surveillance de seuil reste active.

[0038] Un objet "ntP100AlarmClearValue" représente un second seuil de surveillance à atteindre pour émettre une notification dite notification de fin d'alerte après qu'un premier seuil ait été dépassé, premier seuil pour lequel une notification dite notification d'alerte a été émise. Les notifications émises par l'agent de performance sont soit des notifications d'alerte soit des notifications de fin d'alerte. Une valeur "sans" signifie qu'il n'y a pas de notification de fin d'alerte. La valeur par défaut est "défaut", qui correspond à une valeur de fin d'alerte calculée. Par exemple, si le type de comparaison est ">", la valeur par défaut "défaut" est "ComparisonValue - ComparisonValue/10 - 1". Dans ce cas, si l'activité CPU descend en dessous de 80% et qu'une notification d'alerte d'une activité supérieure à 90% a déjà été émise, une notification de fin d'alerte est émise. Cet objet est utile pour ne pas conserver actives un trop grand nombre de notifications d'alertes.

[0039] Un objet "ntP100AlarmCheckPeriod" représente un entier qui permet d'émettre une notification même quand un seuil n'est pas atteint. La valeur "0" par défaut implique aucune notification, "1" implique une notification à chaque fois que le seuil n'est pas atteint, "n" implique une notification si le seuil n'a pas été atteint "n" fois consécutives. "n" est un entier positif.

[0040] Un objet "ntP100AlarmDate" définit la date de création de la surveillance de seuil par une requête SNMP SET.

[0041] L'agent de performance comporte un objet "ntP100AlarmEventLog" qui détermine le type de notification émise. La valeur par défaut est "faux" et implique que la notification sera un message d'alarme "TRAP". La valeur "vrai" implique que la notification sera une écriture dans l'historique du système. Une autre valeur implique que la notification sera une écriture dans un fichier historique portant le nom de ladite autre valeur.

[0042] La table d'alarme comporte également un objet supplémentaire, "ntP100AlarmProgram", qui contient le nom du programme qui doit être exécuté conformément à l'invention quand le seuil est atteint. L'objet « ntP100AlarmProgram » peut prendre trois valeurs. La première valeur "sans" signifie ne pas appeler de programme en cas de détection de dépassement de seuil. La seconde valeur possible est le nom d'un programme à appeler en cas de détection de dépassement de seuil. Cette valeur peut être suivie par d'éventuels paramètres. Une troisième valeur, également en cas de détection de dépassement, peut être le nom de l'agent SNMP de performance (2i). Par exemple, ce sera l'agent décrit ci-dessus "ntPerformance". Cette valeur sera suivie par d'autres valeurs, donnant le nom d'une fonction de diagnostic interne à appeler et les éventuels paramètres de cette fonction ou des indexes d'une table de diagnostic. L'appel de cette fonction permet de déclencher des diagnostics sur un seuil de consommation globale.

[0043] La figure 2 représente une table de seuil (3i). Chaque ligne de la table constitue une entrée de la table qui comprend un ensemble de valeurs des objets (seuils, attributs, formules) de ladite table définissant une requête surveillance de seuil. La première colonne « AlarmID » de la table comprend, par exemple, les valeurs de l'objet "ntP100AlarmID" constituant les indexes des seuils de surveillance et les points d'entrée dans cette table. L'annexe 1 représente un exemple de table de seuil, constituée de requêtes de surveillance de seuil référencées de A1-1 à A1-7.

[0044] Considérons le premier exemple de réalisation de requête de surveillance de seuil de la table de seuil suivant :

- ntP100AlarmFormula 100 - ( DELTA ( ntP01TotalProcessorTime ) / ( DELTA ( ntP01CaptureTime ) \* 100000 ) )
- ntP100AlarmComparisonType >
- ntP100AlarmComparisonValue 90
- ntP100AlarmProgram ntPerformance diag -gc -gm -gs -gn -pc 20-dt 4000 -pm 2000

[0045] Dans cet exemple, si l'agent SNMP de performance (2i) détecte un taux de consommation du ou des pro-

cesseurs de l'ordinateur supérieur à 90 %, premièrement, une notification de détection de dépassement de seuil serait émise vers le gestionnaire SNMP ; deuxièmement, la fonction interne (5i) de diagnostic, " diag ", avec les paramètres -gc -gm -gs -gr -pc 20 -dt 4000 -pm 2000, serait appelée ; troisièmement, la fonction interne (5i) enverrait une notification vers le gestionnaire SNMP contenant ces différentes consommations, avec la date. Ensuite, suivant la valeur de l'objet 5 SNMP "ntP100AlarmEventLog", la fonction pourrait écrire dans l'historique des événements du système, ou dans un fichier historique. L'ordre des paramètres et le nombre de paramètres de la fonction n'a aucune importance. A titre d'exemple, le premier paramètre " -gc " s'il est utilisé signifie que la fonction calculerait la consommation globale en processeur. Le second paramètre " -gm ", s'il est utilisé, signifie que la fonction calculerait la consommation globale de la mémoire physique. Le troisième paramètre " -gs ", s'il est utilisé, signifie qu'elle calculerait la consommation globale 10 de la mémoire du cache. Le quatrième paramètre " -gn ", s'il est utilisé, signifie qu'elle calculerait la consommation globale du réseau. Le cinquième paramètre " -pc 20 ", s'il est utilisé, signifie que cette fonction rechercherait tous les programmes qui utiliseraient un pourcentage paramétrable, par exemple plus de 20 %, du ou des processeurs. Le sixième paramètre de consommation totale dans un délai " -dt 4000 ", s'il est utilisé, signifie que cette fonction diagnostiquerait toutes ces consommations sur un délai paramétrable, par exemple de 4000 millisecondes. Le septième pa- 15 ramètre " -pm 2000 ", s'il est utilisé, signifie que la fonction rechercherait tous les programmes qui utiliseraient plus d'une valeur paramétrable, par exemple plus de 2000 kilo octets, de mémoire physique. La table de seuil peut contenir la destination "ntP100AlarmEventLog" des notifications émises par l'agent de performance. La nature de la notification (message d'alarme ou écriture dans un historique) est ainsi facilement paramétrable. L'objet "ntP100AlarmEventLog" peut être modifié afin que cette notification puisse être constituée d'un message d'alarme ou d'une écriture dans un 20 historique.

[0046] Considérons le deuxième exemple de réalisation de requête de surveillance de seuil d'une table d'alarme suivant :

```

- ntP100AlarmFormula 100 - ( DELTA ( ntP01TotalProcessorTime ) / ( DELTA ( ntP01CaptureTime ) * 100000 ) )
- ntP100AlarmComparisonType >
- ntP100AlarmComparisonValue 90
- ntP100AlarmProgram ntPerformance diag 1 4 23 24 25

```

[0047] Ce deuxième exemple illustre un second mode de réalisation de l'invention. Ce deuxième exemple diffère du premier exemple de réalisation par le fait que les paramètres de la fonction interne ntPerformance 'diag', ne sont plus 30 comme dans l'exemple précédent des arguments -gc -gm -gs -gn -pc appelées par la fonction diag, mais des points d'entrée dans une table de diagnostic "ntP101Diag", et plus précisément des indexes d'accès aux différents diagnostics possibles contenus dans une table de diagnostic (4i) gérable elle aussi par le gestionnaire SNMP. Cette table de diagnostic "ntP101Diag", accepte les formules avec un compteur multi-instancié et permet plus de souplesse en per- 35 mettant son enrichissement par le protocole SNMP. Elle ne surveille pas les seuils contrairement à la table de seuil (3i) "ntP100Alarm", mais établit des diagnostics relatifs au dépassement de seuil. La table de diagnostic interroge les compteurs une seule fois après dépassement du seuil de surveillance. La figure 3 représente un exemple de table de diagnostic (4i) selon l'invention. La structure de cette table de diagnostic est similaire à celle de la table de seuil (3i). Chaque ligne de la table de diagnostic constitue un point d'entrée de la table et comprend un ensemble de valeurs 40 des objets de ladite table définissant une requête de diagnostic de seuil.

[0048] Considérons l'exemple de requête de diagnostic d'index 23 (point d'entrée 23, ligne 23) de la table de dia- gnostic suivant :

```

- ntP101Diagformula ( DELTA ( ntP08ProcessTimePID:* ) / ( DELTA ( ntP08CaptureTime ) * 100000 ) )
- ntP101DiagComparisonType >
- ntP101DiagComparisonValue 10
- ntP101DiagPeriod 4000
- ntP101DiagSeverity avertissement
- ntP101DiagID 23

```

[0049] Les 6 champs possibles de cette table de diagnostic (4i) ont la même signification que leurs équivalents dans la table de seuil. La période, "ntP101DiagPeriod", est en millisecondes.

[0050] Cet exemple diagnostique quels sont les programmes actifs ou processus qui ont consommé chacun plus de 10 % du temps de tous les processeurs. Le caractère "\*" désigne dans cet exemple tous les programmes actifs ou 55 processus et est appelée la star convention. Un mécanisme de cache pour la star convention est ici inutile car les diagnostics sont appelés de manière ponctuelle. Ceci implique aussi que la période de diagnostic peut être courte. La surveillance n'a lieu qu'au niveau des ressources globales. Par contre les diagnostics peuvent être lancés sur des unités d'exécution ("threads") différentes. Si l'ordinateur possède plusieurs processeurs, les différentes unités d'exé-



cution "threads" peuvent être exécutées en parallèle. De façon similaire à la table de seuil, la table de diagnostic peut contenir la destination "nP101DiagEventLog" des notifications émises par l'agent de performance. Cette notification peut être constituée d'un message d'alarme ou d'une écriture dans un historique.

[0051] La surveillance des performances s'effectue par des unités d'exécution qui sont lancées pour chaque surveillance ou activité à surveiller. A chaque requête de seuil et à chaque requête de diagnostic de seuil dépassé, une unité d'exécution est associée. Cette unité d'exécution 'thread' peut être créée, modifiée ou suspendue. Il existe un nombre maximum d'unité d'exécution d'une part pour la surveillance de seuil et d'autre part pour le diagnostic de seuil. Une fonction de création d'une unité d'exécution 'thread' teste si le nombre maximum d'unité d'exécution est atteint et lance l'unité d'exécution. Une requête de surveillance de seuil est modélisée dans une structure qui définit les valeurs de chaque objet de la table de seuil. Cette structure, correspondant à un point d'entrée (une ligne) de la table de seuil, contient (figure 2):

« -son identificateur	'AlarmID'
-sa formule	'AlarmFormula'
-sa période	AlarmPeriod'
-son type de comparaison	'AlarmComparisonType'
-sa valeur à comparer	'AlarmComparisonValue'
-son nombre d'occurrence avant alerte	'AlarmRepeat'
-sa sévérité	'AlarmSeverity'
-son nombre d'alarmes possibles	'AlarmOnce'
-la destination	'AlarmEventLog'
-son programme à appeler	'AlarmProgram' ('Diag')
-sa date	'AlarmDate'
-sa valeur de fin d'alerte	'AlarmClearValue'
-sa période d'activation	'AlarmCheckPeriod'
>>	

[0052] De la manière identique, une requête de diagnostic de seuil est modélisée dans une structure correspondant à un point d'entrée (une ligne) de la table de diagnostic qui contient :

« -son identificateur	'DiagID'
-sa formule	'DiagFormula'
-sa période	'DiagPeriod'
-son type de comparaison	'DiagComparisonType'
-sa valeur à comparer	'DiagComparisonValue'
-sa sévérité	'DiagSeverity'
-sa destination	'DiagEventLog'
-sa date	'DiagDate' »

[0053] Deux structures identiques, internes à une unité d'exécution contiennent les valeurs de compteurs définies dans les formules et mémorisés, la première valeur à un temps donné, la seconde valeur à un temps ultérieur. La différence des deux temps correspond à la période. Les compteurs délivrent des valeurs à des temps donnés ou dans le cas de valeurs trop importantes, des différences de valeurs calculées au bout de ladite période. Ces structures indiquent, premièrement si la différence des compteurs doit être calculée et deuxièmement sur quelle instance de compteur de surveillance le calcul doit être réalisé, dans le cas d'un compteur à multiples instances. En référence à la figure 4, une structure interne (6) d'une unité d'exécution d'une surveillance de seuil mémorise pour chaque compteur (Cs1, Cs2, ..., Csi, ..., Csn), un nom (61), une valeur (62) (correspondant à une valeur donnée à un instant déterminé ou à une différence de valeurs calculées sur la période), un attribut (63) de calcul de différence déterminant selon sa valeur si le calcul de la différence doit être effectué ou non et un index (64) définissant l'objet physique ou logique que le compteur surveille, par exemple le disque 1. En référence à la figure 5, une structure interne (7) d'une unité d'exécution de diagnostic de seuil comporte un nom (71), une valeur (72), et un attribut (73) de calcul de différence des compteurs (Cd1, Cd2, ..., Cdi, ..., Cdn). Les différences des compteurs sont calculés avant le calcul de la formule.

[0054] La première tâche d'une unité d'exécution de surveillance de seuil, correspondant à la requête de surveillance de seuil (correspondant à un point d'entrée dans la table de surveillance) est d'analyser sa formule "AlarmFormula", c'est-à-dire de préparer ses deux structures internes (6) à l'aide de sa formule en mémorisant tous les compteurs

(Cs1,..., Csi ..., Csn) qu'elle doit gérer. La deuxième tâche de l'unité d'exécution est d'obtenir, à un temps donné, les valeurs des compteurs de la première structure interne afin d'initialiser sa surveillance. La troisième et dernière tâche de l'unité d'exécution consiste en une boucle infinie de surveillance. Cette boucle peut être divisée en trois parties. La première partie de la boucle consiste à attendre pendant une durée correspondant à sa périodicité (AlarmPeriod'), puis à obtenir à ce nouveau temps donné les valeurs (valeurs ou différences de valeur) des compteurs de surveillance de la seconde structure interne. La seconde partie de la boucle consiste à tester, par un calcul à l'aide des deux structures internes, si le seuil a été atteint et son nombre d'occurrence (AlarmRepeat) avant l'émission d'une notification d'alerte. Si c'est le cas, la notification est émise avec sa sévérité vers sa destination et son programme à appeler est exécuté. Si le programme à appeler est une série de diagnostic interne (Diag), une unité d'exécution par diagnostic est créée. Si son nombre d'alarme possible est 1, alors l'unité d'exécution de surveillance de seuil est arrêtée. Le compteur associé à sa période de vérification 'CheckPeriod' est remis à zéro. Si le seuil n'a pas été atteint, deux cas se présentent. Premier cas, si une notification a été émise, et si sa valeur de fin d'alerte est atteinte, une notification avec sa sévérité de fin d'alerte est émise. Second cas, si sa période de vérification est atteinte, la notification avec sa sévérité de vérification est émise. La troisième et dernière partie de la boucle consiste à remplacer les anciennes valeurs des compteurs de surveillance de la première structure interne par ceux de la seconde.

**[0055]** La première tâche d'une unité d'exécution de diagnostic, correspondant à la requête de diagnostic, est d'analyser sa formule (DiagFormula) c'est-à-dire de préparer ses deux structures internes (7) à l'aide de sa formule. Il doit aussi rechercher si à un compteur de diagnostic a été associée la chaîne de caractères : \*. Cette chaîne indique qu'il doit rechercher le nombre d'occurrences de ce compteur. La deuxième et dernière tâche d'une unité d'exécution de diagnostic consiste en une boucle finie de surveillance. Elle est finie sur le nombre d'occurrences trouvé ci-dessus. La première partie de la boucle consiste à trouver, à un temps donné, les valeurs de compteur de la première structure interne, afin d'initialiser le diagnostic. La seconde partie de la boucle consiste à attendre pendant une durée correspondant à sa périodicité, puis à obtenir sur ce nouveau temps donné les valeurs des compteurs de surveillance de la seconde structure interne. La troisième partie de la boucle consiste à tester si le seuil a été atteint, par un calcul à l'aide des deux structures. L'unité d'exécution calcule la différence entre les deux compteurs et calcule la formule avec cette différence. Si le seuil est atteint, la notification de diagnostic avec sa sévérité est émise.

**[0056]** La table de seuil et la table de diagnostic peuvent être entièrement gérées à partir de requêtes SNMP SET émises par le gestionnaire SNMP. Les tables sont conservées dans des fichiers qui sont relus au lancement de l'agent de performance. Des fichiers permettant de surveiller de la manière la plus économique tout en ayant des diagnostics les plus précis possibles sont livrés par défaut.

**[0057]** D'autres modifications à la portée de l'homme de métier font également partie de l'esprit de l'invention.

## ANNEXE 1

5	ntP100AlarmID 1 ntP100AlarmFormula $100 - ( \text{DELTA} ( \text{ntP01TotalProcessorTime} ) / ( \text{DELTA} ( \text{ntP01CaptureTime} ) * 100000 ) )$ ntP100AlarmPeriod 3600 ntP100AlarmComparisonType > ntP100AlarmComparisonValue 90 ntP100AlarmRepeat 0 ntP100AlarmSeverity warning ntP100AlarmOnce false 1 ntP100AlarmEventLog true ntP100AlarmProgram ntPerformance diag 1 6 10 19 ntP100AlarmDate 1997/06/06-10:16:36 ntP100AlarmClearValue default ntP100AlarmCheckPeriod 0	A
20	ntP100AlarmID 4 ntP100AlarmFormula $( \text{DELTA} ( \text{ntP01TotalDPCTime} ) / ( \text{DELTA} ( \text{ntP01CaptureTime} ) * 100000 ) )$ ntP100AlarmPeriod 3600 ntP100AlarmComparisonType > ntP100AlarmComparisonValue 30 ntP100AlarmRepeat 0 ntP100AlarmSeverity warning ntP100AlarmOnce false ntP100AlarmEventLog true ntP100AlarmProgram ntPerformance diag 4 16 20 33 ntP100AlarmDate 1997/06/06-10:16:36 ntP100AlarmClearValue default ntP100AlarmCheckPeriod 0	A1-2
35	ntP100AlarmID 5 ntP100AlarmFormula $( \text{DELTA} ( \text{ntP08ProcessorTimePID:1} ) / ( \text{DELTA} ( \text{ntP08CaptureTime} ) * 100000 ) )$ ntP100AlarmPeriod 3600 ntP100AlarmComparisonType < ntP100AlarmComparisonValue 5 ntP100AlarmRepeat 0 ntP100AlarmSeverity warning ntP100AlarmOnce false ntP100AlarmEventLog true ntP100AlarmProgram ntPerformance diag 4 5 15 18 ntP100AlarmDate 1997/06/06-10:16:36 ntP100AlarmClearValue default ntP100AlarmCheckPeriod 0	A1-3
45	ntP100AlarmID 8 ntP100AlarmFormula $( ( \text{DELTA} ( \text{ntP01TotalInterruptssec} ) ) / ( \text{DELTA} ( \text{ntP01CaptureTime} ) ) )$ ntP100AlarmPeriod 3600 ntP100AlarmComparisonType > ntP100AlarmComparisonValue 1000 ntP100AlarmRepeat 0 ntP100AlarmSeverity warning ntP100AlarmOnce false ntP100AlarmEventLog true ntP100AlarmProgram ntPerformance diag 4 6 10 13 ntP100AlarmDate 1997/06/06-10:16:36	A1-4
55		

	ntP100AlarmClearValue default	
	ntP100AlarmCheckPeriod 0	
5	ntP100AlarmID 17	A 1-5
	ntP100AlarmFormula ( ntP01ProcessorQueueLength )	
	ntP100AlarmPeriod 3600	
	ntP100AlarmComparisonType >	
	ntP100AlarmComparisonValue 10	
10	ntP100AlarmRepeat 0	
	ntP100AlarmSeverity warning	
	ntP100AlarmOnce false	
	ntP100AlarmEventLog true	
	ntP100AlarmProgram ntPerformance diag 2 6 11 13	
15	ntP100AlarmDate 1997/06/06-10:16:36	
	ntP100AlarmClearValue default	
	ntP100AlarmCheckPeriod 0	
20	ntP100AlarmID 19	A1-6
	ntP100AlarmFormula ( DELTA ( ntP21Networkutilization:1 ) / ( DELTA ( ntP21CaptureTime ) * 10000 ) )	
	ntP100AlarmPeriod 3600	
	ntP100AlarmComparisonType >	
	ntP100AlarmComparisonValue 40	
	ntP100AlarmRepeat 0	
	ntP100AlarmSeverity warning	
25	ntP100AlarmOnce false	
	ntP100AlarmEventLog true	
	ntP100AlarmProgram ntPerformance diag 4 9 12 13	
	ntP100AlarmDate 1997/06/06-10:16:36	
	ntP100AlarmClearValue default	
30	ntP100AlarmCheckPeriod 0	
35	ntP100AlarmID 21	A1-7
	ntP100AlarmFormula ( ( DELTA ( ntP03Pagessec ) ) / ( DELTA ( ntP03CaptureTime ) ) )	
	ntP100AlarmPeriod 3600	
	ntP100AlarmComparisonType >	
	ntP100AlarmComparisonValue 10	
	ntP100AlarmRepeat 0	
	ntP100AlarmSeverity warning	
	ntP100AlarmOnce false	
	ntP100AlarmEventLog true	
40	ntP100AlarmProgram ntPerformance diag 4 7 11 13	
	ntP100AlarmDate 1997/06/06-10:16:36	
	ntP100AlarmClearValue default	
45	ntP100AlarmCheckPeriod 0	

## Revendications

1. Dispositif permettant de surveiller l'activité d'au moins un ordinateur distant (OD1, ..., OD<sub>i</sub>, ..., OD<sub>n</sub>) à partir d'un gestionnaire SNMP (1) installé sur un ordinateur local (OL) à travers un réseau (10), caractérisé en ce qu'il comprend un agent de performance (2i) installé dans chaque ordinateur distant (OD<sub>i</sub>) apte à détecter le dépassement d'un seuil de consommation d'une ou plusieurs ressources de l'ordinateur distant (OD<sub>i</sub>) à partir d'une fonction de seuil (8i), gérable par le gestionnaire SNMP (1) de l'ordinateur local (OL), fonction qui permet de paramétrer l'agent de performance en lui indiquant des seuils déterminés pour chaque activité surveillée, et d'émettre une notification de dépassement de seuil et en ce que l'agent de performance (2i) exécute au moins un diagnostic lorsqu'un seuil a été dépassé en appelant une fonction de diagnostic (5i) interne à l'agent SNMP de performance (2i), fonction gérable par le gestionnaire SNMP (1) de l'ordinateur local (OL), et émet une notification de diagnostic comprenant

une information de diagnostic.

2. Dispositif selon la revendication 1, caractérisé en ce que les paramètres de la fonction interne de seuil et de diagnostic sont des indexes d'accès à une table respectivement de seuil (3i) et de diagnostic (4i) de l'agent de performance (2i).
3. Dispositif selon la revendication 2, caractérisé en ce que la table de seuil (3i) est constituée des objets suivants : un identificateur, une formule, une période, un type de comparaison, une valeur à comparer (un seuil), un nombre d'occurrence avant alerte, une sévérité, un nombre d'alarmes possibles, une fonction interne de diagnostic à appeler en cas de dépassement de seuil, une date, une valeur de fin d'alerte, une période d'activation.
4. Dispositif selon la revendication 2, caractérisé en ce que la table de diagnostic (4i) est constituée des objets suivants : un identificateur, une formule, une période, un type de comparaison, une valeur à comparer, une sévérité, une date.
5. Dispositif selon l'une des revendications 2 à 4, caractérisé en ce que la table de seuil (3i) et la table de diagnostic (4i) comportent un objet définissant la destination de la notification de dépassement de seuil et un objet définissant la destination de la notification de diagnostic.
6. Dispositif selon l'une des revendications précédentes, caractérisé en ce que la notification est constituée d'un message d'alarme (TRAP) ou d'une écriture dans un historique de son propre programme ou dans un historique global du système.
7. Dispositif selon l'une des revendications précédentes, caractérisé en ce que chaque activité surveillée est effectuée par une unité d'exécution qui déclenche une autre unité d'exécution pour chaque diagnostic.
8. Dispositif selon la revendication 1, caractérisé en ce que la fonction de diagnostic (5i) comporte quatre paramètres "-gc, -gm, -gs, -gn", permettant de déclencher respectivement le calcul de la consommation globale en processeur, de la mémoire physique, de la mémoire du cache, du réseau, un cinquième paramètre "-pc 20", permettant de déclencher la recherche de tous les programmes qui utilisent plus d'un pourcentage paramétrable du ou des processeurs, un sixième paramètre "-dt 4000" permettant de déclencher le calcul des consommations sur un délai paramétrable en millièmes de secondes, et un septième paramètre "-pm 2000" permettant de déclencher la recherche de tous les programmes qui utilisent plus d'un nombre paramétrable de kilo octets de mémoire physique.
9. Procédé d'optimisation de surveillance de seuils de consommation paramétrables sur un ordinateur distant (OD) caractérisé en ce qu'il comporte les étapes suivantes :
  - une étape de surveillance d'un seuil par une fonction interne de seuil (8i);
  - une émission, par l'ordinateur distant (OD) sur lequel l'agent de performance (2i) effectue la surveillance, d'une notification de dépassement de seuil;
  - une étape d'exécution de diagnostics par une fonction de diagnostic interne (5i);
  - une étape d'émission d'une notification contenant le diagnostic associé au dépassement de seuil.
10. Procédé d'optimisation de surveillance selon la revendication 9, caractérisé en ce que l'étape de surveillance de seuil comporte trois tâches, une première d'analyse d'une formule de la fonction de seuil, une deuxième d'obtention à un temps donné des valeurs (62) de compteurs d'une première structure interne (6) et une troisième d'exécution d'une boucle infinie de surveillance.
11. Procédé d'optimisation de surveillance selon la revendication 10, caractérisé en ce que la boucle infinie de surveillance comprend une première partie consistant à attendre sur une période et obtenir au bout de la période les valeurs des compteurs d'une seconde structure, une deuxième partie consistant à tester à l'aide des deux structures internes si un seuil de la fonction de seuil a été atteint et si le nombre d'occurrences est atteint, et une troisième partie consistant à remplacer les valeurs des compteurs de la première structure interne par ceux de la seconde.
12. Procédé d'optimisation de surveillance selon la revendication 9, caractérisé en ce que l'exécution d'un diagnostic comporte l'analyse d'une formule en préparant les valeurs de compteurs (Cd1, Cdn) de diagnostic de deux structures internes (7), la recherche du nombre d'occurrence des compteurs de diagnostic et l'exécution une boucle

de surveillance pour chaque occurrence.

13. Procédé d'optimisation de surveillance selon la revendication 12, caractérisé en ce que, la boucle comprend une première partie consistant à trouver, à un temps donné, la valeur des compteurs de la première structure interne, afin d'initialiser le diagnostic, une deuxième partie consistant à attendre sur sa périodicité, puis à obtenir sur ce nouveau temps donné les valeurs des compteurs de surveillance de la seconde structure interne et une troisième partie consistant à tester à l'aide des deux structures si le seuil a été atteint.

5

10

15

20

25

30

35

40

45

50

55

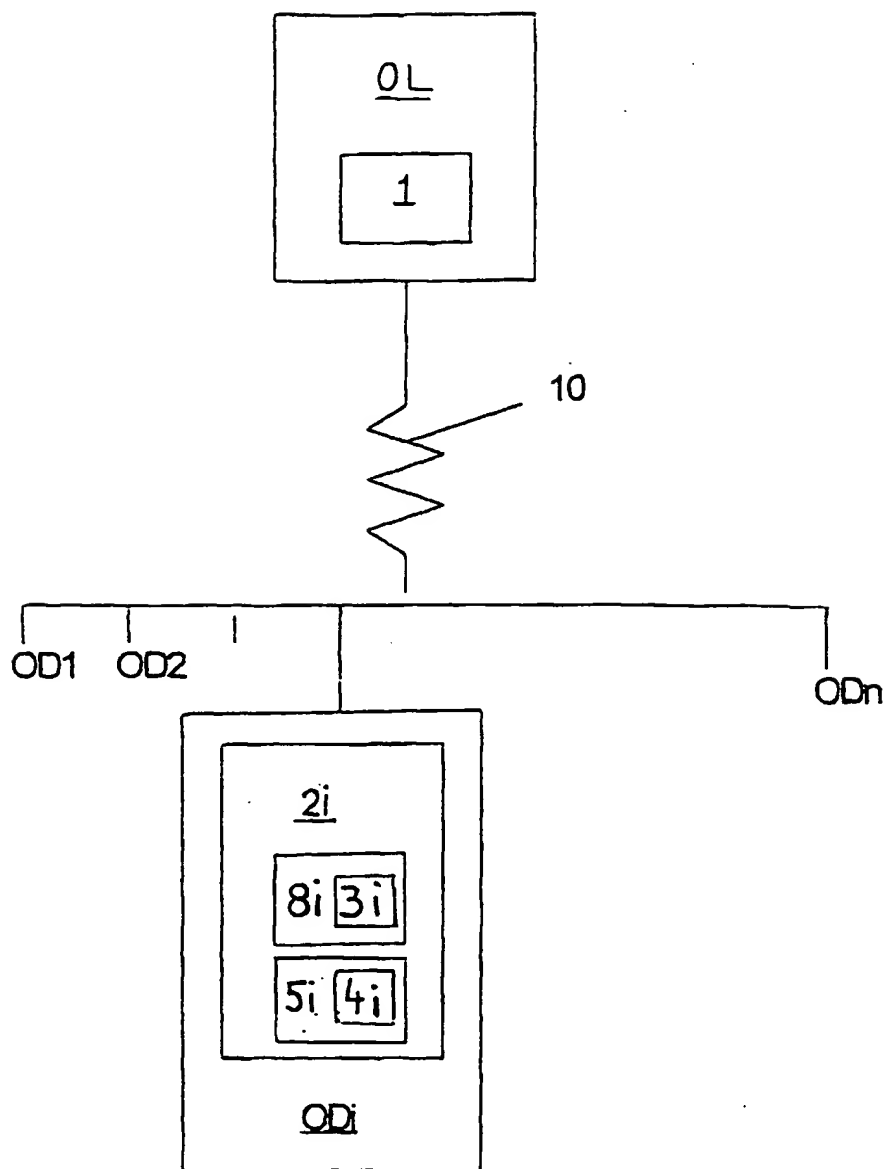


Figure 1

Figure 2

Alarm ID	Alarm Formula	Alarm Period	Alarm Comparison Type	Alarm Comparison Value	Alarm Repeat	Alarm Severity	Alarm OnceAlarm	Alarm EventLog	Alarm Program	Alarm Date	Alarm ClearValue	Alarm Check Period
1	niPFreeSpace : 1	3600	<	3	0	warning	false	TRAP	Diag		default	0
2	niP02Processortime : 1	3600	>	90	0	warning	false	historique	Diag		default	2

3i

Ls1  
Ls2  
Lsn

Figure 3

Diag ID	Diag Formula	Diag Period	Diag Comparison Type	Diag Comparison Value	Diag Severity	Diag EventLog	Diag Date
1	niP08ProcesstimePID : *	4000	>	10	warning	TRAP	

4i

Ld1  
Ldn

Figure 4

Cs1	nom	61
	valeur	62
	Δ	63
	index	64
Cs2	nom	
	valeur	
	Δ	
	index	
Csn		

6

Figure 5

Cd1	nom	71
	valeur	72
	Δ	73
Cd2	nom	
	valeur	
	Δ	
Cdn		

7





Office européen  
des brevets

# RAPPORT DE RECHERCHE EUROPEENNE

Numéro de la demande  
EP 99 40 2940

DOCUMENTS CONSIDERES COMME PERTINENTS			
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes	Revendication concernée	CLASSEMENT DE LA DEMANDE (Int.Cl.7)
X	EP 0 822 498 A (BULL SA) 4 février 1998 (1998-02-04)	1,6-9	606F11/34
Y	<ul style="list-style-type: none"> <li>* page 2, ligne 44 - page 3, ligne 9 *</li> <li>* page 3, ligne 23 - ligne 29 *</li> <li>* page 4, ligne 23 - ligne 27 *</li> <li>* page 4, ligne 46 - page 5, ligne 5 *</li> <li>* page 21, ligne 9 - ligne 10 *</li> <li>* page 34, ligne 47 - ligne 51 *</li> </ul>	2-5, 10-13	
A	EP 0 820 013 A (BULL SA) 21 janvier 1998 (1998-01-21)	1,6-9	
Y	<ul style="list-style-type: none"> <li>* colonne 2, ligne 51 - colonne 3, ligne 2 *</li> <li>* colonne 8, ligne 1 - ligne 55 *</li> <li>* colonne 11, ligne 36 - ligne 43 *</li> <li>* colonne 13, ligne 49 - ligne 54 *</li> <li>* colonne 15, ligne 37 - ligne 38 *</li> </ul>	2-5, 10-13	
			DOMAINES TECHNIQUES RECHERCHES (Int.Cl.7)
			606F
Le présent rapport a été établi pour toutes les revendications			
Lieu de la recherche <b>LA HAYE</b>		Date d'achèvement de la recherche <b>25 février 2000</b>	Examineur <b>Renault, S</b>
CATEGORIE DES DOCUMENTS CITES		T : théorie ou principe à la base de l'invention E : document de brevet antérieur, mais publié à la date de dépôt ou après cette date D : cité dans la demande L : cité pour d'autres raisons & : membre de la même famille, document correspondant	
X : particulièrement pertinent à lui seul Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie A : artère-plan technologique O : divulgation non-écrite P : document interstaté			

EPO FORM 1503 (03/02) (P44003)

**ANNEXE AU RAPPORT DE RECHERCHE EUROPEENNE  
RELATIF A LA DEMANDE DE BREVET EUROPEEN NO.**

EP 99 40 2940

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche européenne visé ci-dessus.  
Lesdits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du  
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets.

25-02-2000

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
EP 0822498 A	04-02-1998	FR 2750517 A	02-01-1998
		CA 2209304 A	27-12-1997
		JP 10091482 A	10-04-1998
EP 0820013 A	21-01-1998	FR 2751448 A	23-01-1998
		US 6021437 A	01-02-2000

EPO FORM P0480

Pour tout renseignement concernant cette annexe : voir Journal Officiel de l'Office européen des brevets, No.12/82